DYNAMIC PARAMETERS CERTIFICATION FOR ASYNCHRONOUS FSM

Nicolae Galupa, PhD
Istanbul Aydın University, Faculty of Engineering, Turkey
Technical University Iasi, Faculty of Automatic Control and Computer Engineering, Romania

Abstract

Although the use of asynchronous sequential machines is confined to solving specific problems where synchronous machines are non-applicable or perform poorly, we can encounter many situations where we definitely wish to exploit their benefits (high speed, low resolution time,…, etc.). However considering the fact that these asynchronous sequential machines are integrated with synchronous machines, a minimal output signal width must be provided in order to obtain the needed control capability. A control method for the output signal width is presented and experimental results confirm its validity. This method represents in fact a hybrid asynchronous model.

Keywords: Asynchronous sequential systems, state, graph, Mealy machine, Moore machine

Foreword

There are situations when synchronous sequential systems are non-applicable. Those situations include:

- input variables may change at any moment of time and they cannot be synchronized with respect to the system's clock;
- the sequential system is implemented using very fast logic gates. In this case the propagation delays along the connection lines became significant inducing clock skewing;
- in case of synchronous sequential systems, the response time in the system's evolution, with respect to the clock signal, is one clock period (fixed response time). However when using asynchronous sequential systems the evolution towards the next state as a response to a change in the input vector is very fast offering very low delays. This is why asynchronous sequential systems are used when designing and implementing high speed structures.

The output of an asynchronous system is active (with respect to time) for a length that is of the same rank with the transition time (Mealy machine) or the response time (Moore machine) - situation that has been presented by Valachi & Bârsan (1984) and Valachi & Hoza (1993), and we have already seen that the output signal's length in time (Z) is proportional with the propagation time through an elementary gate. Onofrei & Valachi (2000) present a method for controlling the output signal's length has been suggested, by using a controlled delay circuit (presetable/programmable delay generator) to generate a signal later used as output when the actual output signal of the asynchronous circuit becomes active (M). The structure suggested is presented in fig.1 (Moore machine).

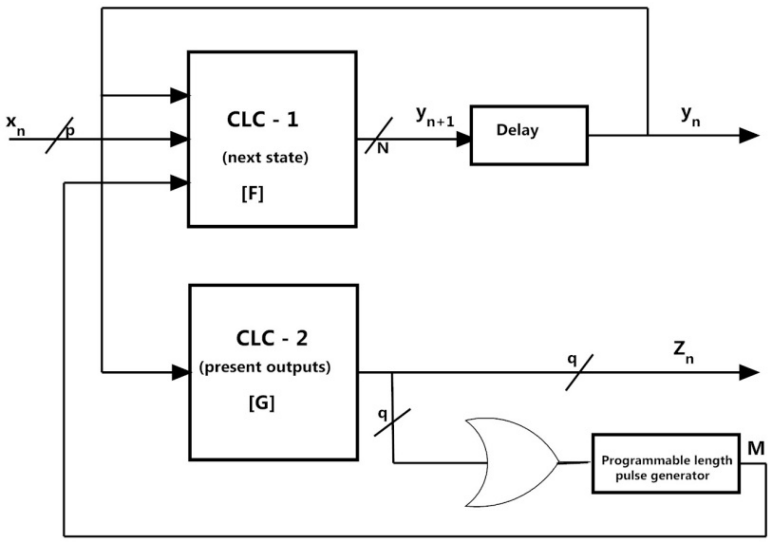


Fig.1

Obviously the system's equations are:

$$Y_{n+1}=F[X_n, M_n, Y_n]; \quad (1.1.)$$

$$Z_n=G[Y_n]; \quad (1.2.)$$

where:

$X_n=\{x_{p-1}, x_{p-2}, \ldots, x_0\}n$ is the input vector;

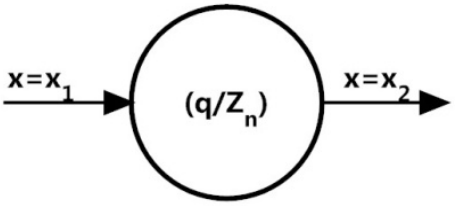
$Y_n=\{y_{N-1}, y_{N-2}, \ldots, y_0\}n$ is the state vector;

$Z_n=\{z_{q-1}, z_{q-2}, \ldots, z_0\}n$ is the output vector;

M_n–supplementary output vector with components triggered by the components of Z_n

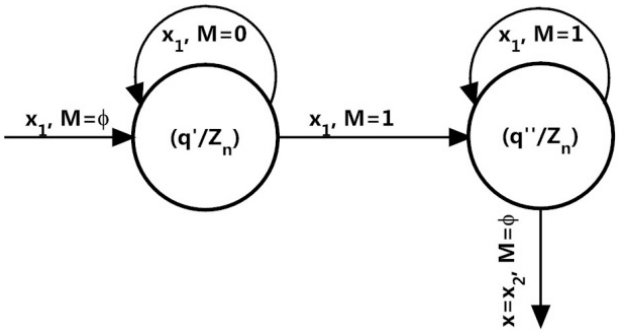
Let's consider an individual state from the system's transition graph, state where at least one element of the output vector is active ($Z_n \neq 0$ – fig 2.a),.

Fig.2.a.



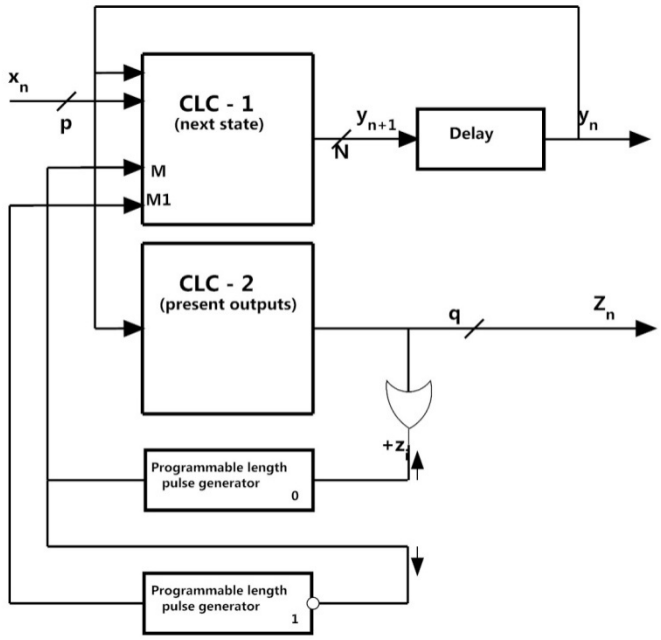
As stated before we cannot guarantee a minimal length in time for the output signal as the transition towards the next state may occur after a time dependent of the technology used (in general very short time – same magnitude level as the propagation time through the gates used). So the solution to certify a minimal length for the output signal would be to replace the state with its equivalent two states as shown in fig. 2.a. Please observe that in this case we are using the M variable that is triggered by the specific output signal we want to enhance.

Fig.2.b.



So we shall double the number of states when we activate au output signal. Also as presented by Onofrei & Valachi (2000), after deactivating the supplementary signal M (1↓0) we cannot instantly reactivate it. A small delay of minimum 10% of the length of M is requested before reactivating this signal. This request is complicating furthermore the structure because we have to use once again a supplementary variable marked by M_1 in this situation. Variable M_1 is activated (M_1 - 0 ↑1) by the negative edge of M (1↓0), and has a constant length in time. During the period of time when M_1 is active the output vector Z is invalidated. The structure that complies with these requests is presented in fig 3.

Fig.3.



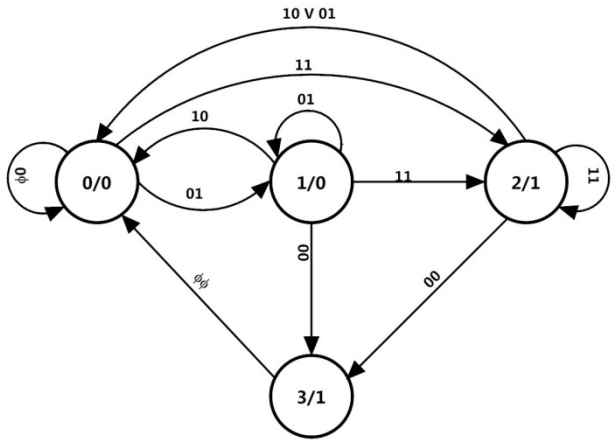
Please notice that this method becomes more and more complicated and difficult to use, as one state that has an active output will be replaced – in this case – by four equivalent states that operate with the supplementary signals M and M_1.

Hazard vs. Output signal relationship

There are situations when the previously presented method for output signal control in an asynchronous circuit is inapplicable because the presence of hazards (Static) at the output of the circuit implementing the asynchronous system. We mean by this that there are situations when the output according to the states transitions graph should not change it's value but during the transition process we experience a short glitch at the output of the circuit, glitch that will trigger the pulse generators if connected as presented in fig 1 or fig 3. We shall make our point by aid of an example.

Let there be the asynchronous sequential system described by the transition graph depicted in fig.4. We have noted by x_1x_0 the components of the input vector and we marked in each state the ratio Q_n/Z_n where Q is the state number and z is the output variable.

Fig.4.



Assuming the following state encoding (Q0→00, Q1→01, Q2→11, Q3→10) and by using the synthesis methods presented in [1], [2] we find:

x_1x_0	$y_1y_{0\,n+1}$			
y_1y_0	00	01	11	10
00	00	01	11	00
01	10	01	11	00
11	10	00	11	00
10	00	00	00	00

x_1x_0	$y_{1\,n+1}$			
y_1y_0	00	01	11	10
00	0	0	1	0
01	1	0	1	0
11	1	0	1	0
10	0	0	0	0

x_1x_0	$y_{0\,n+1}$			
y_1y_0	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	0	1	0
10	0	0	0	0

x_1x_0	z_n			
y_1y_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

And the equations describing the system will be:

$$y_{1,n+1}=\overline{\overline{(y_0\bar{x}_1x_0}\;\overline{\bar{y}_1y_0x_1x_0)}}_n;\quad y_{0,n+1}=\overline{(\overline{\bar{x}_0\;\bar{y}_1y_0x_1)}}_n;\quad z_n=y_{1,n} \tag{2.1.}$$

or

$$y_{1,n+1}=\overline{\overline{(y_0\overline{x_1\oplus x_0}}\;\overline{\bar{y}_1x_1x_0)}}_n;\quad y_{0,n+1}=\overline{(\overline{\bar{x}_0\;\bar{y}_1y_0x_1)}}_n;\quad z_n=y_{1,n} \tag{2.2.}$$

A possible implementation of the circuit is presented in fig. 5

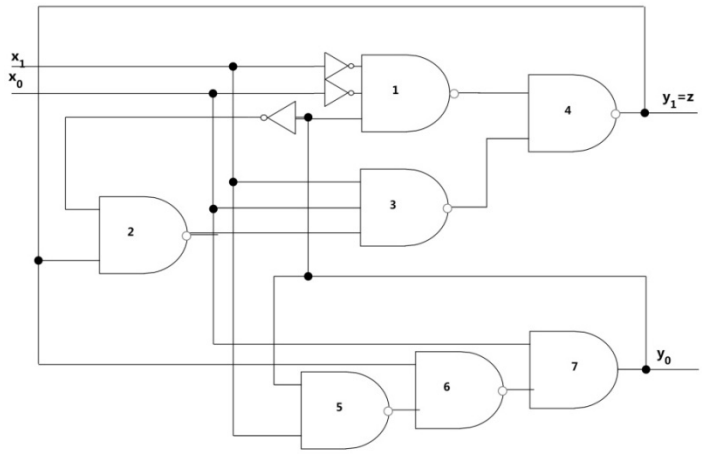


Fig.5.a.
(Eq.2.1. implementation)

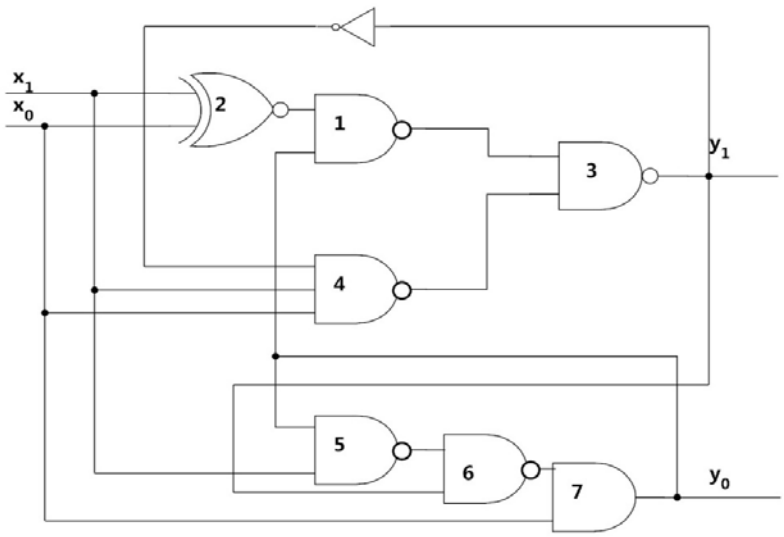


Fig.5.b. (Eq.2.2. implementation)

Let's assume that the system is characterized by x_1x_0=01 and Q=1 (y_1y_0=01). Please notice that, according to the transition graph, the system is in a stable state. For presentation purposes we shall assume that all gates used are characterized by the some propagation time (t_p).

Now suppose that in this situation the input vector changes as follows x_1x_0=01 → x_1x_0=00. According to the transition graph the system should evolve towards state 3 with the output signal z=1. However if we consider the origin of the time axis to be the moment when the input vector changes, after t_p (through gate 7) y_0 will switch from "1" to "0" and that will lock gate 1 so y_1 will not change to "1" but rather remain stuck on "0". In this situation the system will evolve directly towards state 0 (y_1y_0=01) and the output will not be triggered.

Observe that both circuits (different implementation of the same asynchronous sequential system) will respond identically to this specific change on its inputs.

In this case, concerning this specific implementation of the asynchronous sequential machine, we encounter a timing fault that prevents us from using a preprogrammed delay to implement the system's output signal – **we need another solution that would not be so sensitive**.

Output time length control using a synchronizing input

The input vector of the system is extended by adding a new variable S, which in fact is a synchronizing input. We shall use this new input variable as follows – the output signal (when necessary) should be forced active for the time length when S is "high". It is preferably (and we assume

that this is the case) that the new synchronizing input S should have a duty factor of 50% and the time length between two consecutive positive edges is at least one magnitude level higher than the propagation time through the asynchronous structure. By using S the transition graph presented in fig 4 will be equivalently transformed to the transition graph presented in fig 6.a. (input vector x_1x_0S)

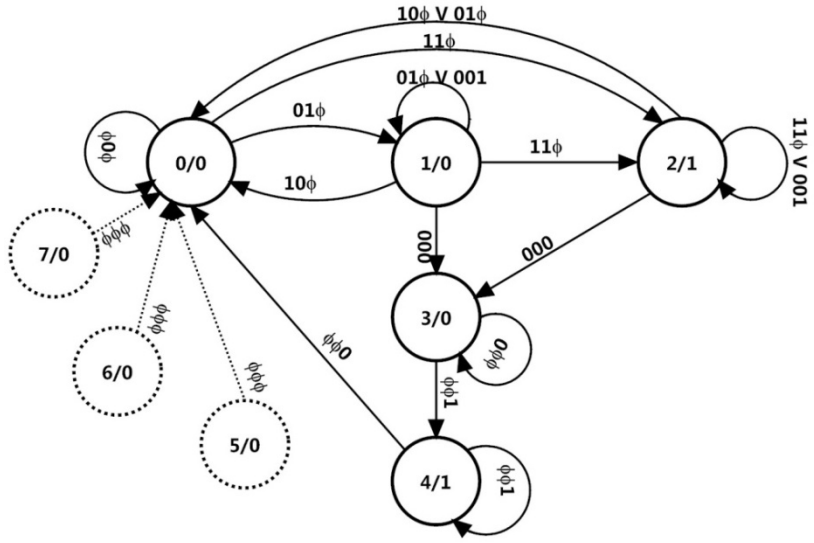


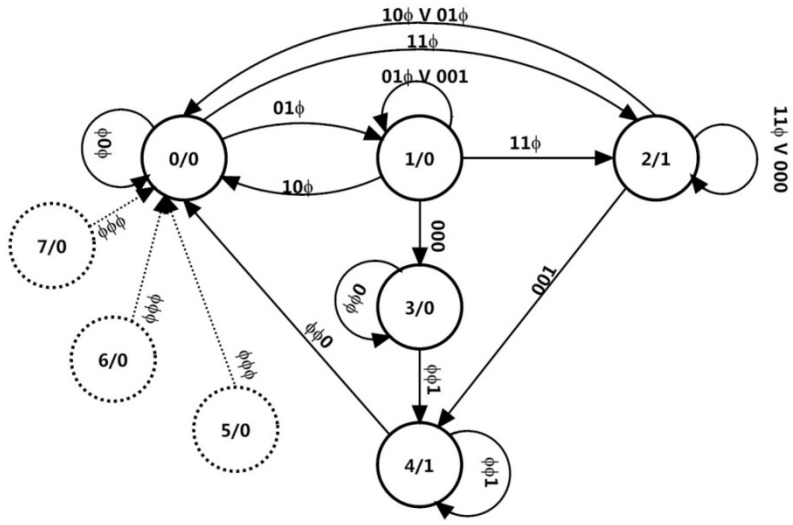
Fig.6.a.

If the output vector is not active (z=0) than the value of S is indifferent to us. However S's value is coherent only when we have at least one output signal active. In the transition graph presented in fig 6.a. we distinguish two interesting situations:

- From state 1 we evolve state 3 only on the negative edge of S and we stay in this state for as long as S is "0". When we reach the positive edge of S the system shall evolve towards state 4 where the output is activated (z=1) and we maintain this situation for as long as S is "1". When the negative edge of S occurs the system will evolve towards the initial state (according to initial transition graph). Now we can see how the output signal has been controlled in length by aid of S when crossing states 1-3-4-0.
- State 2 - length of the output signal is determined by the period of time when both input variables are "1" (x_1x_0=11). In this situation, once again, the S's value is indifferent to us (remember the main reason for using an additional input variable is to certify a minimum length with respect to time for the system's outputs and assuming that the input variables are changing value at a normal rate - larger than t_p - then this condition is fulfilled). However should both input

variables switch to "0" synchronously (meaning the system should evolve to state 3 according to the graph in fig. 4 then we wait for the negative edge of S then apply the same procedure as presented above. Please observe the fact that S is not synchronous with the state transitions that characterize the asynchronous sequential system so when approaching the transition in this manner (Q=2 → Q=3) the output will exhibit z="0" for an unpredictable period of time between the two z="1" situations that are associated with states 2 and 4. If we wish a continuous transition (Q=2 → Q=3) with respect to the output – meaning we don't want an unpredictable length z="0" to show on the output, then the equivalent graph is the one presented in fig.6.b.

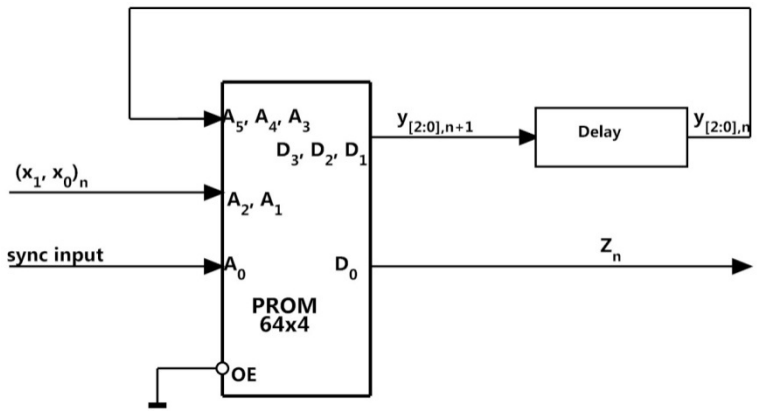
Fig.6.b.



Please observe that the transition graph has been completed with all the transitions and states possible even if indifferent to us (dotted lines). All these states will evolve towards the initial state 0 regardless of the input vector value.

Implementation of such structures with SSI circuits is difficult and not optimal. This is why we strongly suggest the system's implementation to be done using memory circuits as presented by Valachi & Hoza (1993). Such an implementation is presented in fig. 7

Fig.7.



We have used a 32 bytes PROM (64x4). It's address lines are controlled by the input vector and the state vector. The data outputs of the circuit implement the next state vector (the state vector towards which the system evolves) and the output signal. Because in this situation the propagation time from present state towards the next state is defined only by the propagation time through the PROM circuit additional delays are requested. Those delays can be implemented by aid of see through buffers (or even by aid of R-C integrators although this is a solution we do not recommend). The memory contents are defined by the states transition graph of the system. For the system described by fig.6.a. and fig.6.b. the state encoding is:

(Q0→000, Q1→001, Q2→010, Q3→011, Q4→100, Q5→101, Q6→110, Q7→111)

Case 1–(graph in fig.6.a.) → memory content

Binary Address $A_5\ A_4\ A_3\ A_2\ A_1\ A_0$ $(y_2\ y_1\ y_{0)n}\ (x_1\ x_{0)n}\ S_n$	Hex Address	Binary Data $D_3\ D_2\ D_1\ D_0$ $(y_2\ y_1\ y_0)_{n+1} z_n$
0 0 0 0 0 Φ	00 – 01	0 0 0 0
0 0 0 0 1 Φ	02 – 03	0 0 1 0
0 0 0 1 0 Φ	04 – 05	0 0 0 0
0 0 0 1 1 Φ	06 – 07	0 1 0 0
0 0 1 0 0 0	08	0 1 1 0
0 0 1 0 0 1	09	0 0 1 0
0 0 1 0 1 Φ	0A – 0B	0 0 1 0
0 0 1 1 0 Φ	0C – 0D	0 0 0 0
0 0 1 1 1 Φ	0E – 0F	0 1 0 0
0 1 0 0 0 0	10	0 1 1 1
0 1 0 0 0 1	11	0 1 0 1
0 1 0 0 1 Φ	12 - 13	0 0 0 1
0 1 0 1 0 Φ	14 – 15	0 0 0 1
0 1 0 1 1 Φ	16 – 17	0 1 0 1
0 1 1 Φ Φ 0	18,1A,1C, 1E	0 1 1 0

Case 2–(graph in fig.6.b.)→memory content

Binary Address $A_5\ A_4\ A_3\ A_2\ A_1\ A_0$ $(y_2\ y_1\ y_{0)n}\ (x_1\ x_{0)n}\ S_n$	Hex Address	Binary Data $D_3\ D_2\ D_1\ D_0$ $(y_2\ y_1\ y_0)_{n+1} z_n$
0 0 0 0 0 Φ	00 – 01	0 0 0 0
0 0 0 0 1 Φ	02 – 03	0 0 1 0
0 0 0 1 0 Φ	04 – 05	0 0 0 0
0 0 0 1 1 Φ	06 – 07	0 1 0 0
0 0 1 0 0 0	08	0 1 1 0
0 0 1 0 0 1	09	0 0 1 0
0 0 1 0 1 Φ	0A – 0B	0 0 1 0
0 0 1 1 0 Φ	0C – 0D	0 0 0 0
0 0 1 1 1 Φ	0E – 0F	0 1 0 0
0 1 0 0 0 0	10 *	0 1 0 1
0 1 0 0 0 1	11 *	1 0 0 1
0 1 0 0 1 Φ	12 - 13	0 0 0 1
0 1 0 1 0 Φ	14 – 15	0 0 0 1
0 1 0 1 1 Φ	16 – 17	0 1 0 1
0 1 1 Φ Φ 0	18,1A,1C,1E	0 1 1 0

0 1 1 Φ Φ 1	19,1B,1D, 1F	1 0 0 0	0 1 1 Φ Φ 1	19,1B,1D,1F	1 0 0 0
1 0 0 Φ Φ 0	20,22,24,2 6	0 0 0 1	1 0 0 Φ Φ 0	20,22,24,26	0 0 0 1
1 0 0 Φ Φ 1	21,23,25,2 7	1 0 0 1	1 0 0 Φ Φ 1	21,23,25,27	1 0 0 1
1 0 1 Φ Φ Φ	28 – 2F	0 0 0 0	1 0 1 Φ Φ Φ	28 – 2F	0 0 0 0
1 1 Φ Φ Φ Φ	30 – 3F	0 0 0 0	1 1 Φ Φ Φ Φ	30 – 3F	0 0 0 0

Conclusion:

- The present paper suggests an asynchronous sequential system design methodology that uses an extra synchronizing input variable that will certify minimal dynamic parameters for the output signals of the systems (whenever active).

- We prove the output signal control using presetable length pulse generators is not efficient because it complicates the design and it induces a larger number of states to deal with.

- By inserting an extra input variable for synchronizing purposes we notice that we can eliminate the propagation hazards caused by the regular input variables.

- If the number of states when the output variables are active is small the overall speed of such a system is close to the speed of a totally asynchronous sequential system

References:

Valachi Al. & Bârsan M. (1984) *Numerical Technique and Automata*, Ed. Junimea Iasi,

Valachi Al. & Hoza Fl. (1993) *Analysis, Synthesis and Testing of Digital Devices*, Ed. Nord-Est Iasi,

Onofrei V. & Valachi Al. (2000) Control Method for the output signals width in asynchronous systems.
Iasi Polytechnic Institute Bulletin, Tom XLI(L), Fasc 1-4